II. AMENDMENTS TO THE CLAIMS

The following listing of claims replaces all prior versions, and listings, of claims in the application:

1. (Currently Amended) A method for detecting software development best practice violations, comprising:

receiving sets of source code from a plurality of sources;

extracting at least one code pattern from the sets of source code;

defining meta data for each of the at least one code pattern that indicates a quality of the at least one code pattern;

assigning a rank to each of the at least one code pattern based on the corresponding meta data,

storing each of the at least one code pattern and the assigned rank in a data structure;

receiving a subsequent set of source code having no affiliation with, and ~~that is~~ developed independently from[[,]] all of the at least one code pattern stored in the data structure;

extracting and classifying a code pattern to be tested from the subsequent set of source code;

comparing the code pattern to be tested to the at least one code pattern stored in the data structure to determine a closest match to the code pattern to be tested;

assigning a rank of the closest match to the code pattern to be tested; and

detecting a software development best practice violation if the rank assigned to the code pattern to be tested fails to comply with a predetermined threshold.

2. (Original) The method of claim 1, wherein the rank is further based on a skill level and an experience level of a developer of the at least one code pattern.

3. (Original) The method of claim 1, further comprising determining a programming language of the sets of source code.

4. (Original) The method of claim 3, wherein the meta data for each of the at least one code pattern identifies the programming language, a list of most used classes, a list of dependencies, a number and a type of objects created and used at run time, and memory usage of the at least one code pattern.

5. (Original) The method of claim 1, wherein the plurality of sources comprises a plurality of nodes interconnected in a peer-to-peer network environment, and wherein each of the plurality of nodes is operated by a developer.

6. (Original) The method of claim 5, further comprising registering the developers, prior to the receiving step.

7. (Original) The method of claim 6, wherein the registering step comprises collecting contact information, a skill level and an experience level corresponding to the developers.

8. (Original) The method of claim 7, wherein the registering step further comprises collecting feedback information about each developer from the other developers.

9. (Canceled).

10. (Currently Amended) A method for building a dynamic best practice violation (BPV) engine resource for detecting software development best practice violations, comprising:

receiving sets of source code from a plurality of sources;

detecting a programming language of each of the sets of source code;

extracting a plurality of code patterns from the sets of source code;

defining meta data for each of the plurality of code patterns that indicates a quality of the plurality of code patterns;

classifying and assigning a rank to each of the plurality of code patterns based on the corresponding meta data,

storing each of the plurality of data patterns and the assigned rank in a data structure;

receiving a subsequent set of source code having no affiliation with the plurality of code patterns stored in the data structure;

extracting and classifying a code pattern to be tested from the subsequent set of source code plurality of code patterns in the data structure developed independently from the plurality of code patterns stored in the data structure;

comparing the code pattern to be tested to the plurality of code patterns stored in the data structure to determine a closest match to the code pattern to be tested;

assigning a rank of the closest match to the code pattern to be tested; and

detecting a software development best practice violation if the rank assigned to the code pattern to be tested fails to comply with a predetermined threshold.

11. (Original) The method of claim 10, wherein the plurality of sources comprises a plurality of nodes interconnected in a peer-to-peer network environment, and wherein each of the plurality of nodes is operated by a developer.

12. (Original) The method of claim 11, further comprising registering the developers, prior to the receiving step.

13. (Original) The method of claim 12, wherein the registering step comprises collecting contact information, a skill level and an experience level corresponding to the developers.

14. (Original) The method of claim 13, wherein the registering step further comprises collecting feedback information about each developer from the other developers.

15. (Previously Presented) The method of claim 10, wherein the rank is further assigned to each of the plurality of code patterns based on a skill level and an experience level of a developer of each of the sets of source code.

16. (Original) The method of claim 10, wherein the meta data for each of the plurality of code patterns identifies the programming language, a list of most used classes, a list of dependencies, a number and a type of objects created and used at run time, and memory usage of each of the plurality of code patterns.

17. (Canceled).

18. (Currently Amended) A method for detecting software development best practice violations, comprising:

receiving sets of source code from a plurality of sources;

detecting a programming language of each of the sets of source code;

extracting the plurality of code patterns from the sets of source code;

defining meta data for each of the plurality of code patterns that indicates a quality of the plurality of code patterns; and

assigning a rank to each of the plurality of code patterns based on the corresponding meta data,

storing the plurality of code patterns and the ranks in a best practice violation (BPV) engine resource,

receiving a first set of source code in the BPV engine, the first set of source code having no affiliation with, and ~~been~~ developed independently from[[,]] the plurality of code patterns;

extracting and classifying a code pattern to be tested from the first set of source code;

comparing the code pattern to be tested to a plurality of code patterns extracted from the sets of source code previously received and analyzed by the BPV engine to determine a closest match to the code pattern to be tested;

assigning a rank previously assigned to the closest match to the code pattern to be tested; and

detecting a software development best practice violation if the rank assigned to the code pattern to be tested fails to comply with a predetermined threshold.

19. (Canceled).

20. (Previously Presented) The method of claim 18, wherein the rank is further assigned to each of the plurality of code patterns based a skill level and a experience level of a developer of each of the other sets of source code.

21. (Previously Presented) The method of claim 18, wherein the plurality of sources comprises a plurality of nodes interconnected in a peer-to-peer network environment, and wherein each of the plurality of nodes is operated by a developer.

22. (Original) The method of claim 21, further comprising registering the plurality of nodes, prior to the receiving step.

23. (Original) The method of claim 22, wherein the registering step comprises collecting contact information, a skill level and an experience level corresponding to the developers.

24. (Original) The method of claim 23, wherein the registering step further comprises collecting feedback information about each developer from the other developers.

25-52. (Canceled).